2.22 DETAILED RESULTS FOR DETECTION THRESHOLD

The Detection Threshold Functional Element (FE) was found to be implemented as specified in Section 2.22 of ASP II.

The quality of the code implementing the FE is poor, with the exception of the detection decision implementation. The poor code portion apparently was written before modern structured programming practices were adopted; the code implementing the approximation to the incomplete gamma function should be rewritten using modern software development techniques.

Internal code documentation for the FE is poor. Limited external documentation was available; the User/Analyst/Programmer Manual almost entirely omits description of characteristics related to the FE. The lack of documentation related to the FE is a deficiency resulting in the inability of a prospective user to evaluate threshold modeling in *RADGUNS*.

The table below summarizes the desk checking and software testing verification results for each design element in the Detection Threshold FE. One entry is listed for each design element. The results columns contain check marks if no discrepancies were found during verification. Where discrepancies were found, the Desk Check Result column contains references (D1, D2) to discrepancies listed in Table 2.22-4, while the Test Case Result column identifies the relevant test cases which explain the discrepancies listed in Table 2.22-6.

TABLE 2.22-1. Verification Results Summary for Detection Threshold FE.

Design Element	Code Location	Desk Check Result	Test Case ID	Test Case Result
22-1: Identifying a Threshold Value	PERCUE, 169-195 SRCH1, 349-376	D1	22-1 through 22-6	Y
22-2: Detection Decision	PERCUE, 169-195 SRCH1, 349-376	D2	22-2 through 22-6	Y
22-3: Bias Level Approximation Technique	PRBDET, 61-92 DGAM, DEVAL, SUMLOG	Y	22-8 through 22-12	Y
22-4: False Alarm Number	PRBDET, 43,68	Y	22-7	Y
User Input	INPUT1, INPUT2	Y	all	Y
System Parameters	RDRDAT	Y	all	22-9

2.22.1 Overview

The detection threshold of a radar is the minimum target-signal power for which the radar can detect a target, with a given probability of success, in the presence of the radar thermal noise and/or some external influence. Since the factors influencing target detection are generally noise-like (randomly occurring), the criterion for detection is usually described by some form of probability distribution with associated probabilities of detection (P_d) and false alarm (P_{fa}). Signal integration and signature fluctuations (described in Sections 2.25 and 2.4, respectively) are closely related to this functional element.

Signal-to-Interference ratio (S/I) denotes the ratio of the power of a target return at the radar receiver to the sum of the power due to receiver noise and ground clutter. In a simple S/I detection decision methodology, target detection occurs provided that S/I exceeds a pre-specified value (the threshold). In the next step up in complexity, the P_d is calculated. In this case, detection can be based either on this probability exceeding a certain value, or on a random draw.

In *RADGUNS*, either the S/I or P_d is compared with a user-selectable value to determine detection. The purpose of the Detection Threshold Functional Element in *RADGUNS* is to establish a target detection status.

RADGUNS implements the FE in Subroutines PERCUE, SRCH1, and PRBDET. Subroutine SRCH2 is applicable to the FE, but does not model the applicable AAA system. Modules DGAM, DEVAL, and SUMLOG support PRBDET calculations. These and other subroutines used for this FE are briefly described in Table 2.22-2. None of these routines are dedicated to the Threshold FE; they also implement other FEs (such as RCS, Signal Integration, and Signature Fluctuations). Verification encompassed the portions applicable to the Detection Threshold FE for the subject AAA system.

TABLE 2.22-2. Subroutine Descriptions.

Module Name	Description
PRBDET	Calculates the false alarm number, bias level voltage, and P_d for Swerling Case 1.
DGAM	Main module which calculates approximation to the incomplete gamma function. Uses Subroutine DEVAL.
DEVAL	Calculates one term of the approximation to the incomplete gamma function. Uses Subroutine SUMLOG.
SUMLOG	Accumulates natural logarithms of integers between zero and the module argument.
PERCUE	Simulates radar operation for a perfect cueing search pattern. Also determines target detection status.
SRCH1	Simulates radar operation for sector and circular search patterns. Also determines target detection status.
SRCH2	Simulates radar operation for a circular search pattern. Also determines target detection status.
INPUT1, INPUT2	Reads (from data file) and initializes user input.
RDRDAT	Initializes system-specific variables.

2.22.2 Design Elements

The Detection Threshold FE contains four design elements. A design element is an algorithm that represents a specific component of the FE design. These elements are specified in the Design Approach section of the ASP II for *RADGUNS*. The design elements for the FE are listed in Table 2.22-3.

TABLE 2.22-3. Threshold Design Elements.

Module	Design Element	Description
PERCUE, SRCH1	22-1: Identifying a Threshold Value	Establishes the use of either a P_d "threshold" or a S/I threshold
PERCUE, SRCH1	22-2: Detection Decision	Compares either S/I or P_d with a user-defined threshold to establish the target detection status
PRBDET, DGAM, DEVAL, SUMLOG	22-3: Bias Level Approximation Technique	Computes the value of the incomplete gamma function, which in turn is used in bias level approximations
PRBDET	22-4: False Alarm Number	Computes Fehlner's false alarm number
INPUT1, INPUT2	Input	Reads and initializes user input
RDRDAT	Input	Initializes radar parameters

2.22.3 Desk Check Activities and Results

The code implementing this FE was manually examined using the procedures described in Section 1.1 of this report. Discrepancies found by desk checking are described in the following table.

TABLE 2.22-4. Desk Checking Discrepancies.

Design Element	Desk Check Result
22-1: Identifying a Threshold Value	D1: No problems were encountered based on comparison with ASP II description of the design approach. However, the documented design characteristics are based on a reverse engineering process. No reference is known which corroborates the current algorithm used to determine which threshold values to use. The conclusions based on the observed software are discussed in Section 2.22.5.
22-2: Detection Decision	D2: The ASP II design approach does not require two consecutive received signals to be above the threshold for a target detection declaration, although that is implemented in code. Further, no independent reference is known which corroborates this requirement.

While the code implementing the approximation of the incomplete gamma function is basically correct, the numerical methods are implemented in a very inefficient manner, the programming conventions are poor, and internal documentation is scarce. (According to the header in Subroutine PRBDET, the code was completed in 1971, which was before coding practices were established that are considered good by current standards.) Internal documentation and code quality problems are characterized in the following table. Conclusions and recommendations based on the information in the table are provided in Section 2.22.5, Conclusions and Recommendations.

TABLE 2.22-5. Code Quality and Internal Documentation Results.

Module	Code Quality	Internal Documentation
PRBDET	Many variables are not initialized. No variable declarations are performed. In Subroutine PRBDET, unnecessary reassignments of variables to new variable names occur; variables YB and SNR are copied to variables YS and X, respectively. Also, an unnecessary assignment of KASE+1 to K occurs. Arithmetic IF statements are used abundantly; such statements are not desirable because of the inherent three-way GO-TO characteristic. With one exception, all arithmetic IF statements had only two different branch numbers. Subroutine PRBDET has hundreds of executable lines of code, but less than one-half of the code is accessible. This is because the original author coded P_d calculations for all Swerling Models (0-4), but only Swerling Type 1 is allowed due to the setting of KASE=1 in subroutine PDET which calls PRBDET. The computational GO-TO statement based on the value of KASE always sends execution to only one of the five coded P_d algorithms. Another variable, MODE, is set to one, and then is used in an arithmetic IF statement; thus, only one execution path is possible, so the branch from PRBDET line 42 to line 45 never occurs; this is due to the author choosing one method over another for computing the false alarm number, but does not produce incorrect results.	Internal documentation for the FE is poor. Formatted headers are not included in modules. Most variables are not defined. Only a few short phrases such as “compute bias level” and “Case 1" are interspersed in the code; no descriptions are provided for equations, algorithms, recursive loops, or logic flow. Applies to all modules equivalently.
GAM	Many variables are not initialized. No variable declarations are performed. Variable TN in Function GAM is set, but has no apparent effect on module operation. Arithmetic IF statements are used abundantly; such statements are not desirable because of the inherent three-way GO-TO characteristic. All arithmetic IF statements had only two different branch numbers.	
EVAL, SUMLOG	Many variables are not initialized. No variable declarations are performed. Arithmetic IF statements are used abundantly; such statements are not desirable because of the inherent three-way GO-TO characteristic. All arithmetic IF statements had only two different branch numbers.	

2.22.4 Software Test Cases and Results

Software testing for Subroutines PERCUE, SRCH1, and PRBDET was performed by running the entire *RADGUNS* model. For these tests, *RADGUNS* was run using the subject AAA system, A10A as the target, and EX2.PAR as the parameters file. EX2 was supplied with the *RADGUNS* code. In most cases these files were used as delivered. The Digital Command Language (DCL) file RUNRG.COM was modified to run *RADGUNS* in VAX debug mode. Any other file modifications are listed in the test descriptions. Software testing for DGAM and DEVAL was performed by executing an off-line driver. The values for the number of pulses integrated (variable NPI) and the absolute value of the power of ten of the probability of false alarm (variable PFA), as well as the necessary equations from Subroutine PRBDET, were coded directly into the driver.

Functions DGAM, DEVAL, and SUMLOG solve the approximation of the incomplete gamma function in the Detection Threshold FE. Functions GAM, EVAL, and SUMLOG perform the same function in the Signature Fluctuations FE. The two series solutions are solved by identical algorithms. The differences are slight in the modules utilized by the FEs; these differences have no impact on function values. Section 2.22.5 presents a discussion of the differences.

Software tests for Function DGAM essentially were comparisons with results of GAM using identical input arguments for both modules. Tests of function GAM in Section 2.4, Signature Fluctuations, revealed no discrepancies. Thus, identical answers for both GAM and DGAM was the criteria for establishing verification of DGAM. Similarly, Functions DEVAL and EVAL results were compared for verification purposes. Function SUMLOG is used by both FEs. SUMLOG was verified during Signature Fluctuations FE software testing, thus eliminating the requirement for its testing during threshold verification.

Table 2.22-6 presents the software test cases for the Detection Threshold FE.

TABLE 2.22-6. Software Test Cases for the Detection Threshold FE.

Test Case ID	Test Case Description
22-1	OBJECTIVE: Verify that threshold data are correctly initialized. PROCEDURE: 1. Execute *RADGUNS*, and observe initialization of DETTYP and THRESH(1). 2. Re-run *RADGUNS*, except enter "THRS", "Y", and "10.0 15.0" in the user input parameters file at item #5[1], 5[2], and 5[3], respectively. VERIFY: 1. Variable initializations observed in Step 1 are DETTYP = PDET, and THRESH(1) = 0.5. 2. Variable initialization observed in Step 2 are DETTYP = THRS, THRESH(1) = 10.0, and THRESH(2) = 15.0. RESULT: OK
22-2	OBJECTIVE: Test Design Elements 22-1 and 22-2 for a P_d "threshold". Verify algorithm for establishing a target detection status. PROCEDURE: 1. Enter "PERC" at item #4 of the user input file (use a "perfect cueing" antenna search pattern). 2. Execute *RADGUNS*, and observe initialization of SCHPAT, DETTYP, THRESH(1). 3. Observe whether execution of PERCUE or SRCH1 occurs. 4. After the first calculation of variable SIRAT (in Subroutine PERCUE or SRCH1), deposit a value of 0.9 in SIRAT. 5. Observe whether Subroutine PDET is called in the IF-THEN block following the assignment of SIRAT. 6. Observe whether variable RSTAT is changed to DETECTED. 7. Repeat Steps 2 through 5, except deposit a value of 1.1 in variable SIRAT. 8. Deposit a value in variable PD that is less than the value of THRESH(1). 9. Observe whether variable RSTAT is changed to DETECTED. 10. Repeat Step 7. 11. Deposit a value in variable PD that is greater than the value of THRESH(1). 12. Observe whether variable RSTAT is changed to DETECTED. VERIFY: 1. The initializations observed in Step 2 match the values entered in the user input file. 2. Subroutine PERCUE is called in Step 3 rather than Subroutine SRCH1. 3. The call to PDET in Step 5 does <u>not</u> occur. 4. RSTAT is <u>not</u> changed to DETECTED in Step 6. 5. The call to PDET occurs in Step 7. 6. RSTAT is <u>not</u> changed to DETECTED in Step 9. 7. RSTAT is changed to DETECTED in Step 12. RESULT: OK

TABLE 2.22-6. Software Test Cases for the Detection Threshold FE. (Contd.)

Test Case ID	Test Case Description
22-3	OBJECTIVE: Test Design Element 22-1 and 22-2 for a S/N threshold. Verify the algorithm for establishing a target detection status in a non-cluttered environment for a perfect cueing search pattern. PROCEDURE: 1. Enter "PERC" at item #4 of the user input file. 2. Enter "THRS", "Y", and "10.0 15.0" in the user input parameters file at item #5[1], 5[2], and 5[3], respectively. 3. Execute *RADGUNS* and observe initialization of SCHPAT, DETTYP, PNOISA, THRESH(1), and THRESH(2). 4. Observe whether execution of PERCUE or SRCH1 occurs. 5. Observe the values of variables TS and CL. 6. Freeze program execution immediately after the "ELSE" statement of the IF-THEN block following calculation of variable SIRAT. 7. Deposit a value less than TS/10 into variable CL. 8. Run one additional executable line of code. 9. Deposit a value greater than THRESH(1) x (CL + PNOISA) into variable TS, using the results of Steps 3 and 7. 10. Observe whether variable RSTAT is set to DETECTED. 11. Repeat Steps 3 through 10, except in Step 9, deposit a value less than THRESH(1) x (CL + PNOISA) into variable TS. VERIFY: 1. Values of SCHPAT, DETTYP, THRESH(1) and THRESH(2) observed in Step 3 match the user input data entered in Steps 1 and 2. 2. Execution of Subroutine PERCUE occurs (rather than SRCH1) in Step 4. 3. The detection decision procedures of Steps 7 through 10 occur in the non-cluttered environment IF-THEN block (which uses the value of THRESH(1) for a detection decision). 4. Variable RSTAT is set to DETECTED in Step 10. 5. When performing Step 11, the detection decision procedure (from Steps 8 through 10) is not executed, and thus variable RSTAT is <u>not</u> set to DETECTED. RESULT: OK

TABLE 2.22-6. Software Test Cases for the Detection Threshold FE. (Contd.)

Test Case ID	Test Case Description
22-4	OBJECTIVE: Test Design Element 22-1 and 22-2 for a S/N threshold. Verify the algorithm for establishing a target detection status in a cluttered environment for a perfect cueing search pattern. PROCEDURE: 1. Repeat Steps 1 through 6 from Test Case 22-3. 2. Deposit a value greater than TS/10 into variable CL. 3. Run one additional executable line of code. 4. Deposit a value greater than THRESH(2) x (CL + PNOISA) into variable TS, using the results from Step 1. 5. Observe whether variable RSTAT is set to DETECTED. 6. Repeat Steps 3 through 6 of Test Case 22-3, and Steps 2 and 3 from the present test case. 7. Deposit a value less than THRESH(2) x (CL + PNOISA) into variable TS. 8. Observe whether variable RSTAT is set to DETECTED. VERIFY: 1. The detection decision procedure of Steps 2 though 5 occur in the cluttered environment IF-THEN block (which uses the value of THRESH(2) for a detection decision). 2. Variable RSTAT is set to DETECTED in Step 5. 3. When performing Step 6, the detection decision procedure (from Steps 3 through 5) is not executed, and thus variable RSTAT is <u>not</u> set to DETECTED in Step 8. RESULT: OK
22-5	OBJECTIVE: Test Design Element 22-1 and 22-2 for a S/N threshold. Verify the algorithm for establishing a target detection status in a non-cluttered environment for sector and circular search modes. PROCEDURE: 1. Enter “SECT” at item #4 of the user input file. 2. Repeat Steps 2 through 4 of Test Case 22-3. 3. Freeze program execution immediately following the “ELSE” statement of the IF-THEN block following assignment of variable OPTOUT(7). 4. Deposit a value less than the value of TSMAX/10 into variable CLATTS, and deposit a value greater than THRESH(1) into variable SIRATM. 5. Observe whether variable RSTAT is set to DETECTED. 6. Observe whether variable NDET1 is assigned a value in the IF-THEN block referred to in Step 3. 7. Continue execution through the beginning of the second pass through DO-loop 30, and examine the value of NDET1. 8. Observe the value of NDET2 at the second executable line of DO-loop 30; continue program execution. 9. Repeat Steps 3 through 5. VERIFY: 1. Execution of Subroutine SRCH1 occurs (rather than PERCUE) in Step 2. 2. Variable RSTAT is <u>not</u> set to DETECTED in Step 5. 3. Variable NDET1 is equal to 1 in Steps 6 and 7. 4. Variable NDET2 is equal to 1 in Step 8. 5. Variable RSTAT is set to DETECTED in Step 9. RESULT: OK

TABLE 2.22-6. Software Test Cases for the Detection Threshold FE. (Contd.)

Test Case ID	Test Case Description
22-6	OBJECTIVE: Test Design Element 22-1 and 22-2 for a S/N threshold. Verify the algorithm for establishing a target detection status in a cluttered environment for sector and circular search modes. PROCEDURE: 1. Enter "CIRC" at item #4 of the user input file. 2. Repeat Steps 2 and 3 of Test 22-5. 3. Deposit a value into variable CLATTS that is greater than that of variable TSMAX/10, deposit a value into variable SIRATM that is greater than that of variable THRESH(2), and deposit a value of 1 in variable NDET2. 4. Observe whether variable RSTAT is changed to DETECTED. VERIFY: 1. Variable RSTAT is changed to DETECTED in Step 4. RESULT: OK
22-7	OBJECTIVE: Test Design Element 22-4, False Alarm Number. PROCEDURE: 1. Run *RADGUNS*, and observe the value assigned to variable PFA in Subroutine PDET. 2. Observe the initialization of variable PFA in Subroutine PRBDET. 3. Observe the value of variable ENPR calculated in Subroutine PRBDET at lineÊ68. VERIFY: 1. Value of PFA observed in Step 2 matches that of Step 1. 2. Value of ENPR observed in Step 3 matches independent calculation using ASP II Equation [2.22-24]. RESULT: OK
22-8	OBJECTIVE: Verify the initial bias level estimate. Subroutine PRBDET is the test subject, unless indicated otherwise. PROCEDURE: 1. Run *RADGUNS*, and observe the value assigned to variable NPI in Subroutine PDET. 2. Observe initialization of variable NPI in Subroutine PRBDET. 3. Observe the value of FAN. 4. Observe the value of ENPR at line 57 and EN at line 58. 5. Deposit a value of NPI = 12 after execution of line 59. 6. Observe execution path after line 61. 7. Observe value of YBPR. 8. Repeat Steps 1 through 7, except deposit a value of NPI = 13 at Step 5. VERIFY: 1. Value of NPI observed in Step 2 equals that of Step 1. 2. Value of ENPR observed in Step 4 equals the value of FAN observed in Step 3. 3. Value of EN observed in Step 4 equals the value of NPI observed in Step 2. 4. Execution branches to line 62 in Step 6. 5. Value of YBPR observed in Step 7 matches independent calculation of ASP II Equation [2.22-10], using EN and ENPR from Step 4. 6. Execution branches to line 64 in Step 8. 7. Value of YBPR observed in Step 8 matches independent calculation of ASP II Equation [2.22-11], using observed values of EN and ENPR. RESULT: OK

TABLE 2.22-6. Software Test Cases for the Detection Threshold FE. (Contd.)

Test Case ID	Test Case Description
22-9	OBJECTIVE: Verify correct initialization of variables required before successive approximations of the bias level occur. Subroutine PRBDET is the test subject, unless indicated otherwise. PROCEDURE: 1. Run *RADGUNS*, and observe the initialization of variables NPI, EN, ENPR, and YBPR. 2. Observe the initialization of DGAM input arguments (ENPR, NPI-1) in the call to DGAM at line 69, and note the value of variable GAMPR. 3. Observe the value of variable PYB at line 70. 4. Observe the value of variable SUML at line 71. 5. Deposit a value into variable GAMPR that is greater than the value of variable PYB. 6. Observe the execution path after line 72. 7. Observe the value of variable H. 8. Repeat Steps 1 through 4. 9. Deposit a value into variable GAMPR that is less than the value of variable PYB. 10. Observe the execution path after line 72. 11. Observe the value of variable H. 12. Observe the value of variables Y0 at line 76 and E0 at line 77. VERIFY: 1. Values of input arguments ENPR and NPI initialized in Step 2 match those observed in Step 1. 2. Value observed in Step 3 matches independent calculation of $f(Y_b)$ in ASP II Equation [2.22-8]. 3. Execution transfers to line label 12 in Step 6. 4. Value of H observed in Step 7 equals -0.1. 5. Execution transfers to line label 10 in Step 10. 6. Value of H observed in Step 11 equals 0.1. 7. Value of Y0 observed in Step 12 matches the value of YBPR in Step 1. 8. Value of E0 observed in Step 12 matches independent calculation of ASP II Equation [2.22-18]. RESULT: OK

TABLE 2.22-6. Software Test Cases for the Detection Threshold FE. (Contd.)

Test Case ID	Test Case Description
22-10	OBJECTIVE: Verify the successive approximation technique for determining the bias level. Subroutine PRBDET is the test subject, unless indicated otherwise. PROCEDURE: 1. Run *RADGUNS*, and observe the values of variables GAMPR, PYB, SUML, H, Y0, and E0 before execution of line 78. 2. Observe the value of variable Y1 at line 78. 3. Observe the call to Function DEVAL at line 79, noting the value of its input arguments. Note the value returned by DEVAL. 4. Observe the value of E1 at line 79. 5. Observe the value of variable STEP at line 80. 6. Observe the execution path after line 82. 7. Observe the value of Y0, Y1, and GAMPR at lines 83, 84, and 85 respectively. 8. Observe the execution path after line 86. 9. Allow execution through the recursive loop to proceed until the value of the arithmetic IF statement at line 82 equals zero. 10. Observe the execution path after the condition in Step 9 is met. VERIFY: 1. The value of variable Y1 observed in Step 2 equals the sum of variables Y0 and H observed in Step 1. 2. The value of the first input argument noted in Step 3 matches the value of Y1 observed in Step 2. 3. The returned value of Function DEVAL in Step 3 matches independent calculation of ASP II Equation [2.22-18]. 4. The value of E0 observed in Step 4 matches the value of DEVAL observed in Step 3. 5. The value of STEP observed in Step 5 matches independent calculation of ASP II Equation [2.22-16] using the values of GAMPR, H, E0, and E1 observed in Steps 1 and 4. 6. Execution transfers to line label 18 in Step 6, line label 16 in Step 8, and line label 20 in Step 10. RESULT: OK

TABLE 2.22-6. Software Test Cases for the Detection Threshold FE. (Contd.)

Test Case ID	Test Case Description
22-11	OBJECTIVE: Verify the algorithm for interpolating between the final two bias approximations to identify a final bias level. PROCEDURE: 1. Develop a driver to exercise Subroutine PRBDET by incrementing the value of NPI with each run, while using the following input arguments: SNDB = 10.0; PFA = 8.0; KASE = 1; NPI = 2 2. Run the driver, and freeze execution at the first occurrence of the condition GAMPR > PYB at line 72. 3. Observe the execution path after line 72. 4. Note the value of variable H. 5. Freeze execution at line 87, and note the values of variables Y0, Y1, PYB, STEP, and GAMPR. 6. Observe the execution path to variable YB. 7. Observe the value of YB. 8. Allow execution to continue until the condition GAMPR < PYB occurs at line 72. 9. Repeat Steps 3 through 7. VERIFY: 1. Execution transfers to the line labelled 12 in Step 3. 2. Value of H observed in Step 4 equals -0.1. 3. Execution transfers to the line labelled 22 in Step 6. 4. Value of YB observed in Step 7 matches independent calculation using ASP II Equation [2.22-21] for the case H=-0.1. 5. Execution transfers to the lines labelled 10 and 24 in Step 9. 6. Value of YB observed in Step 9 matches independent calculation using ASP II Equation [2.22-21] for the case H=+0.1. RESULT: OK
22-12	OBJECTIVE: Test Functions DGAM and DEVAL by comparing results with those of GAM and EVAL. PROCEDURE: 1. Develop a driver to call functions GAM and DGAM using input arguments B = 50 and N = 0. Insert a DO-loop in the driver which decrements variable B by 5.0 and increments variable N by 5 before each call to GAM and DGAM. Create a driver termination condition when N equals 50. 2. Run the driver, and observe the values of GAM, DGAM, EVAL, and DEVAL for each pass through the DO-loop. VERIFY: 1. Value of GAM equals the value of DGAM with each pass through the DO-loop. 2. Value of EVAL equals the value of DEVAL with each pass through the DO-loop. RESULT: OK

2.22.5 Conclusions and Recommendations

2.22.5.1 Code Discrepancies

Verification activities revealed few discrepancies in *RADGUNS* v.1.8 implementation of the Detection Threshold FE. The discrepancy is due to no independent reference for the

algorithm used to determine a threshold value. Also, no independent reference was found to support the number of consecutive threshold crossings required to establish a target detection. The sources of these algorithms should be identified and described in the model documentation.

2.22.5.2 Code Quality and Internal Documentation

The code quality of the detection decision performed in Subroutines PERCUE and SRCH1 is good. The code implementing the approximation to the incomplete gamma function basically yields correct results, but the numerical methods are implemented in a very inefficient manner, and the programming conventions are poor. Although the code quality of Subroutine PRBDET generally is poor, the code portion which determines the bias level is good. PRBDET calls Functions DGAM, DEVAL, and SUMLOG. The code quality of Functions GAM, EVAL, and SUMLOG in Section 2.4.5 (FE 4, Signature Fluctuations) also applies to Functions DGAM, DEVAL, and SUMLOG for the Detection Threshold FE. The code quality discussion is repeated here for completeness.

Comprehensive variable initialization should be incorporated in future releases. Also, unused variables and unnecessary variable reassignment should be eliminated. Finally, inaccessible code should be eliminated, and not just changed to comment statements. Storing outdated or unused code in a model release is not recommended.

The extensive code quality and internal documentation problems detailed in Table 2.22-5 applies to modules implementing the probability of detection model (which incorporates the Detection Threshold FE). Alleviating the problems will require more than just adding a few extra comment lines or implementing a few simple code adjustments. Examination of the existing modules indicates that a significant code revision and documentation effort will be required to alleviate the stated problems. Therefore, rewriting the modules using modern structured programming techniques is recommended.

2.22.5.3 External Documentation

The external documentation is non-existent for explanation of false alarm number and bias level calculations in Subroutine PRBDET. In fact, descriptions of PRBDET, DGAM, DEVAL, and SUMLOG (GAM and EVAL also) are entirely omitted from the combined User/Analyst/Programmer Manual for *RADGUNS*.

The combined manual has analyst's information included in appendices that present theoretical descriptions of several modeled processes. However, the methodology for establishing a target detection status in *RADGUNS* is not presented in the manual. Also, the algorithm for calculating the bias level and the false alarm number are omitted from the manual. Design approach information related to the Detection Threshold FE should be explained in a separate appendix.

The programmer's manual information is documented in the form of module descriptions. File RGPDET.FOR contains seven modules which implement the probability of detection model (which span three functional elements, but only four modules apply to the Detection Threshold FE). Descriptions of these modules are omitted entirely from the manual, and thus should be added to Section V, Subroutine and Function Descriptions.